Procedia Computer Science

Volume 51, 2015, Pages 1473–1482

ICCS 2015 International Conference On Computational Science

# A Pipelining Implementation for High Resolution Seismic Hazard Maps Production

Yelena Kropivnitskaya[1], Jinhui Qin[2], Kristy F. Tiampo[1] and Michael A. Bauer[2]

[1]*Earth Sciences Department, Western University, London, Canada*
[2]*Computer Science Department, , Western University, London, Canada*
*ykropivn@uwo.ca, jqin5@uwo.ca, ktiampo@seis.es.uwo.ca, bauer@uwo.ca*

**Abstract**

Seismic hazard maps are a significant input into emergency hazard management that play an important role in saving human lives and reducing the economic effects after earthquakes. Despite the fact that a number of software tools have been developed (McGuire, 1976, 1978; Bender & Perkins, 1982, 1987; Robinson, et al., 2005, 2006; Field, et al., 2003), map resolution is generally low, potentially leading to uncertainty in calculations of ground motion level and underestimation of the seismic hazard in a region. In order to generate higher resolution maps, the biggest challenge is to handle the significantly increased data processing workload.

In this study, a method for improving seismic hazard map resolution is presented that employs a pipelining implementation of the existing EqHaz program suite (Assatourians & Atkinson, 2013) based on IBM InfoSphere Streams – an advanced stream computing platform. Its architecture is specifically configured for continuous analysis of massive volumes of data at high speeds and low latency. Specifically, it treats processing workload as data streams. Processing procedures are implemented as operators that are connected to form processing pipelines. To handle large processing workload, these pipelines are flexible and scalable to be deployed and run in parallel on large-scale HPC clusters to meet application performance requirements. As a result, mean hazard calculations are possible for maps with resolution up to 2,500,000 points with near-real-time processing time of approximately 5-6 minutes.

*Keywords:* Hazard maps, pipelining, stream computing, high performance computing (HPC)

## 1 Introduction

Today, probabilistic seismic hazard analysis (PSHA), based on the total probability theorem (Cornell, 1968; McGuire, 2004), is the most widely used method for estimating seismic hazard analysis. This is particularly true for hazard map calculations, where the probability of a ground

Selection and peer-review under responsibility of the Scientific Programme Committee of ICCS 2015



doi:10.1016/j.procs.2015.05.337

motion acceleration value occurring at any given site is estimated by integrating conditional probabilities over all possible distance and magnitude values. There is a number of methodological considerations in PSHA mapping, and hazard map resolution is one of the most important of those technical issues today (Musson & Henni, 2001). In this context, resolution is defined as the spatial grid resolution that is used to produce map or grid density. A coarse grid results in loss of details for smaller source zones and, as a result, could underestimate the maximum ground motion level value (Musson & Henni, 2001). However, hazard computation with a finer grid requires considerably more computational resources, especially with a complex model. For instance, calculation for more than 6.5 trillion locations is necessary in order to obtain a hazard map for Eastern Canada with a resolution of one dot per meter. Additionally, if a Monte Carlo simulation approach is used for the PSHA calculation (Musson, 1999, 2000), where resolving power improves with the number of simulations used to generate synthetic data, then the processing workload is even heavier and also requires additional computational power. For example, 100,000 simulations of 100 years duration is equivalent to 10,000,000 years of seismicity data. However, significant technological breakthroughs in high performance computing (HPC) have taken place over the last few decades. For example, a cluster of computers can be connected together with advanced networks to provide increased computational power on demand. Complex problems can be partitioned into smaller tasks and programmed into pipelines and deployed on these computer clusters. Heavy workload can be split and distributed onto those pipelines and processed in parallel to meet application performance needs.

IBM InfoSphere Streams (Streams), an advanced stream computing platform, is designed specifically for parallelism and deployment across computing clusters for continuous analysis of massive volumes of data with high speeds and low latency. In particular, Streams treats processing workload as data streams. Processing procedures are implemented as operators that are connected together to form processing pipelines. To handle large processing workload, these pipelines are flexible and scalable. Here, PSHA mapping programs are designed to be deployed and run in parallel on large-scale HPC clusters. The result is better real-time seismic hazard analysis through distributed computing networks.

The main goal of this work is to utilize innovative computational algorithms for PSHA mapping by integrating different input data sources and existing processing tools into a streaming and pipelined computing application. A set of high-resolution maps for different frequencies calculated in terms of the probability of exceeding a certain ground motion level across Eastern Canada are obtained. The motivation for this study is not the estimation of seismic hazard in Eastern Canada, but is to demonstrate that the use of the pipelining and streaming techniques provided by Streams that makes possible the production of high-resolution hazard maps in near-real time with no limitations on resolution. This approach could be used in any region in the world where seismic sources and ground motion characteristics are available. The results can also be used as input for sensitivity analysis of hazard maps for any input models, something that has not been done before, largely because of difficulties with computational implementation.

# 2 Monte Carlo Approach for PSHA Mapping

One of the straightforward and flexible PSHA methodologies involves the use of Monte Carlo simulations (Musson, 1999). Here seismic source models representing the spatial and temporal historical earthquakes occurrence in the region are used as the first stage in Monte Carlo simulations for the generation of a synthetic earthquake catalogue. As a result, the synthetic catalogue shows a set of probable earthquakes in the region over a long time period, on the order of 100 years. In the next stage, this synthetic catalogue is used to estimate a distribution of ground motions at a number of sites using selected ground motion prediction equations (GMPEs) and/or attenuation parameters (Musson, 1999). Finally, probabilities of exceeding a certain ground motion level at every point across a region

are calculated. The maps take into account uncertainties in the earthquake location, size and the resulting ground motions that can affect region.

Today there is a number of both free and commercial software tools available to perform PSHA, including those based on the Monte Carlo simulation approach (EQRISK (McGuire, 1976); FRISK (McGuire, 1978); SeisRisk (Bender & Perkins, 1982, 1987); Fortran codes produced by National Seismic Hazard Mapping Project (NHSMP) from the U.S. Geological Survey (USGS); CRISIS (Ordaz, et al., 2013); EQRM (Robinson, et al., 2005, 2006); OpenSHA (Field, et al., 2003)). The Monte Carlo PSHA tool implemented here is the EqHaz software suite of open-source FORTRAN programs developed by Assatourians and Atkinson (2013). This suite consists of three programs. EqHaz1 creates the synthetic earthquake catalogues generated by the user-specified seismicity parameters. EqHaz2 produces the ground motion catalogues at a site, mean hazard probability curves and mean hazard motions at specified return periods calculated for a site or grid of points. EqHaz3 de-aggregates the hazard, producing the relative contributions of the different earthquake sources as a function of distance and magnitude. The EqHaz suite has a number of limitations which mean that it cannot be applied to hazard calculations with more than 1,000,000 records in each synthetic catalog and for more than 100,000 grid points. These limits are barriers on the production of high-resolution hazard maps.

# 3 Pipelining Implementation in Streams

IBM has been actively involved in streaming concept development, resulting in the IBM InfoSphere Streams product (Streams) (IBM, 2014). Streams is a software package that provides a runtime platform, programming model, and toolkits. Stream Processing Language (SPL) is the programming language used in Streams for building applications in stream processing – a data-centric programming model.

The building blocks of a SPL application are data streams and operators. Data streams consist of data packets (i.e. tuples) structured for user defined schema. Operators define the operations on data streams. Operators and streams are assembled together for each application as a data-flow graph which defines the connections among data sources (inputs), operators, and data sinks (outputs).

The deployment of a Streams application is supported by the underlying runtime system. The Streams runtime model consists of distributed processes. Single or multiple operators form a processing element (PE). The Streams compiler and runtime services make it easy determine where to best deploy those PEs, either on a single machine or across a cluster in order to meet the resource requirements (Bauer, et al., 2010).

The advantage of using these techniques is that it allows for the use of multiple computational units without the need to manage allocation, synchronization or communication among them. In addition, Streams Studio is an integrated development environment based on Eclipse with a Streams plug-in. The Streams programming interface has a set of toolkits of built-in operators (IBM, 2014) that can be used directly to speed up the development work. A subset of Streams built-in operators used in the current implementation is listed in Table 1, in order to better understand the implementation procedure.

In the implementation for PSHA mapping in Streams, the EqHaz1 and EqHaz2 Fortran source codes have been modified slightly and compiled into shared system object libraries and linked into Streams applications. Streams allows for the creation of new primitive operators in a more traditional native language, such as Java™ or C++ (IBM, 2014), that can directly call those PSHA procedures from the shared libraries in the streaming environment. In this application, two primitive operators have been implemented. One is the Catalog operator, which encapsulates EqHaz1 procedures for the catalogue generation. The other is the Map operator, which encapsulates EqHaz2 procedures for the map generation. The pipelined implementation is showed on Figure 1 captured from Streams Studio.

In order to clearly display the entire pipeline it is divided into two stages in the figure. The connection between these stages is shown by circles filled with the same color. The first stage shows the process of input parameters gathering. The second one contains PEs performing PSHA procedures and outputting results. Algorithm 1 in Appendix presents all the associated steps.

| Name | Base explanation | Purpose in the implementation |
|---|---|---|
| InetSource | Periodically checks listed remote (Internet or intranet network) data sources, acquires data and generates a stream from them | To gain all input model parameters needed for analysis, including seismic zones model, ground motion prediction equations (GMPEs) and attenuation parameters |
| Custom | Receive and send any number of streams to perform user-specified analysis on stream | To normalize all input seismic sources and GMPE models from different sources to the same formats |
| Functor | Transforms input tuples into output ones and filter if needed | To filter user-defined parameters for two different stages: catalog generation and ground motion estimation |
| Barrier | Synchronizes tuples from several streams | To synchronize all input parameters before analysis begins |
| Split | Under a user specified condition splits a stream into several output streams | To partition the workload of Monte Carlo simulations of large synthetic catalogs and ground motion estimation on many sites |
| TCPSink | Writes data tuples to a TCP socket | Configured as a TCP server to sink outputs in JSON format |

**Table 1:** Built-in stream operators used in the implementation **(IBM, 2014)**

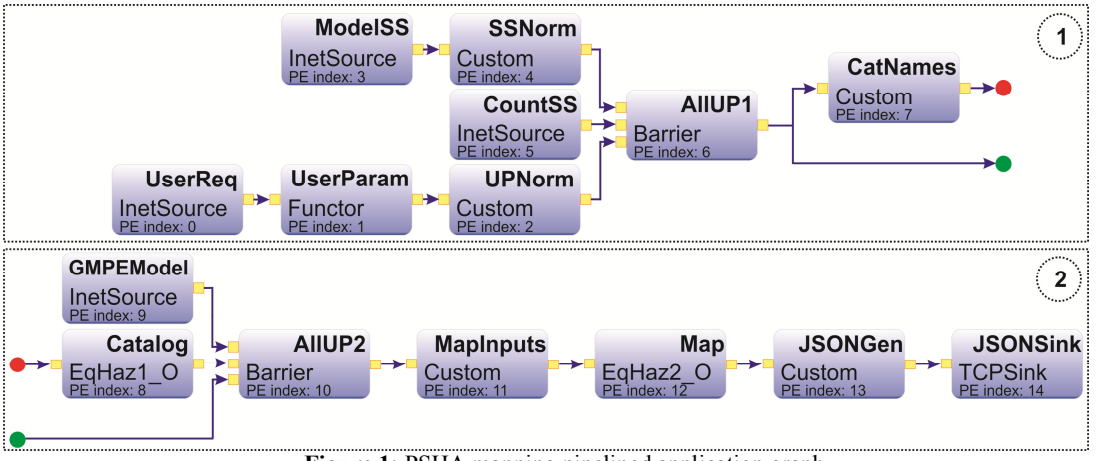


**Figure 1:** PSHA mapping pipelined application graph

All models considered as input streams are located on the Internet at the site www.seismotoolbox.ca. In Algorithm 1, the developed application waits for a new user-request for map generation or for an indication of any updates in the models such that the maps must be updated (Step 1). As soon as this occurs, it generates an input stream for the synthetic catalog generation procedure (Step 2). Afterwards, the stream goes to the catalogue generation primitive operator

developed from the EqHaz1 program (Step 3). Next, the generated synthetic catalog, GMPEs from F4 and other user parameters from F3 are combined to form an input stream for the maps generation module, based on the EqHaz2 program (Steps 4-6). Subsequently, the output stream is converted to JavaScript Object Notation (JSON) format, which is an open standard used to transmit data objects consisting of attribute-value pairs (Step 7). The JSON map file contains grid point coordinates represented by two attributes: "lat" for latitude and "lon" for longitude. The calculated value is represented by spectral acceleration (in our case pseudo acceleration (PSA)) for a given period of time at some level of probability (in this example, 2% in a 50 year return period) at each point of the grid. This is denoted as the "count" attribute. The total output with the set of map coordinates and values sinks to a TCP port (Step 8), where it is taken by a specifically designed JAVA script to be displayed on the webpage. The dynamic gmaps-heatmap.js JavaScript library is used for visualization purposes (Wied, 2014).

As introduced previously, the catalog generation operator has the greatest processing workload because it contains Monte Carlo simulations. The map generation operator also involves a heavy workload if the calculations are required on a dense grid. Therefore, either or both are the bottlenecks of the entire processing scheme for generating high resolution PSHA maps. To overcome EqHaz limitations and to reduce the application execution time, the workload has been decomposed and these two components have been split into multiple pipelines and the executions are parallelized. The complete application graph of the implementation is shown on Figure 2 and Figure 3.

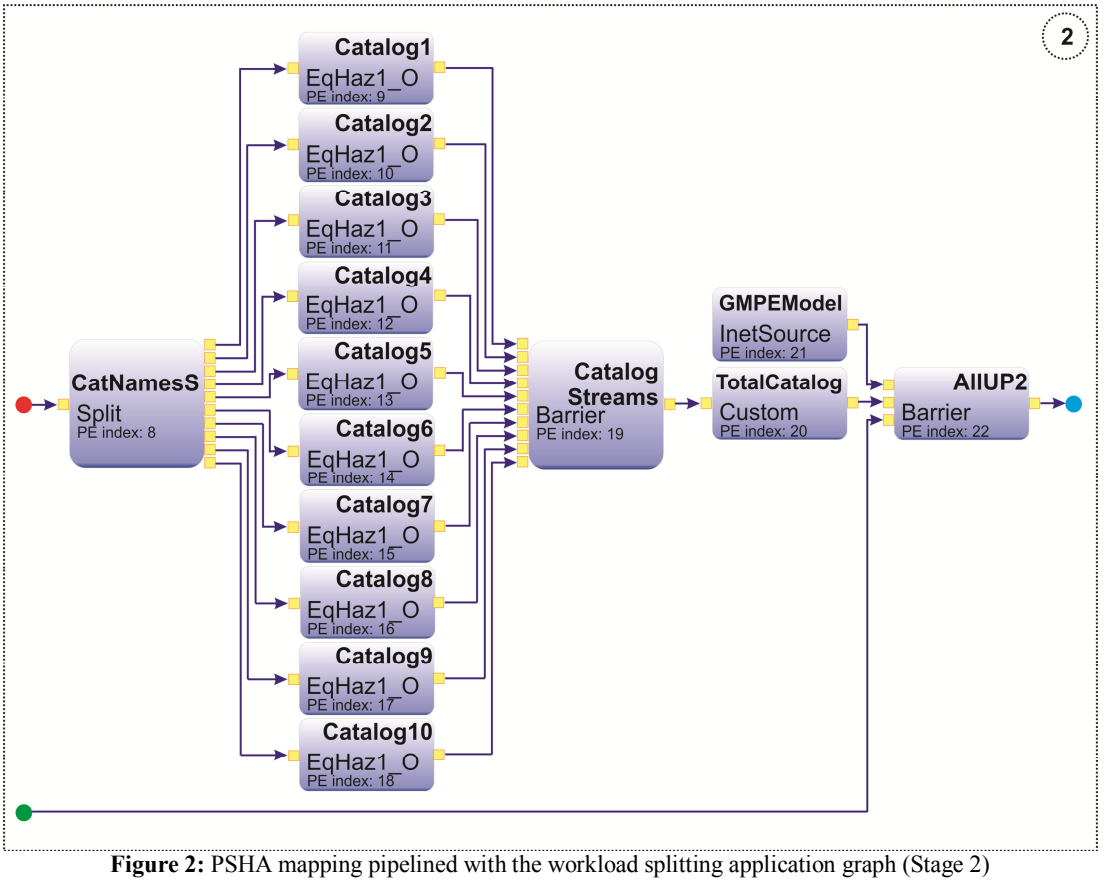


**Figure 2:** PSHA mapping pipelined with the workload splitting application graph (Stage 2)

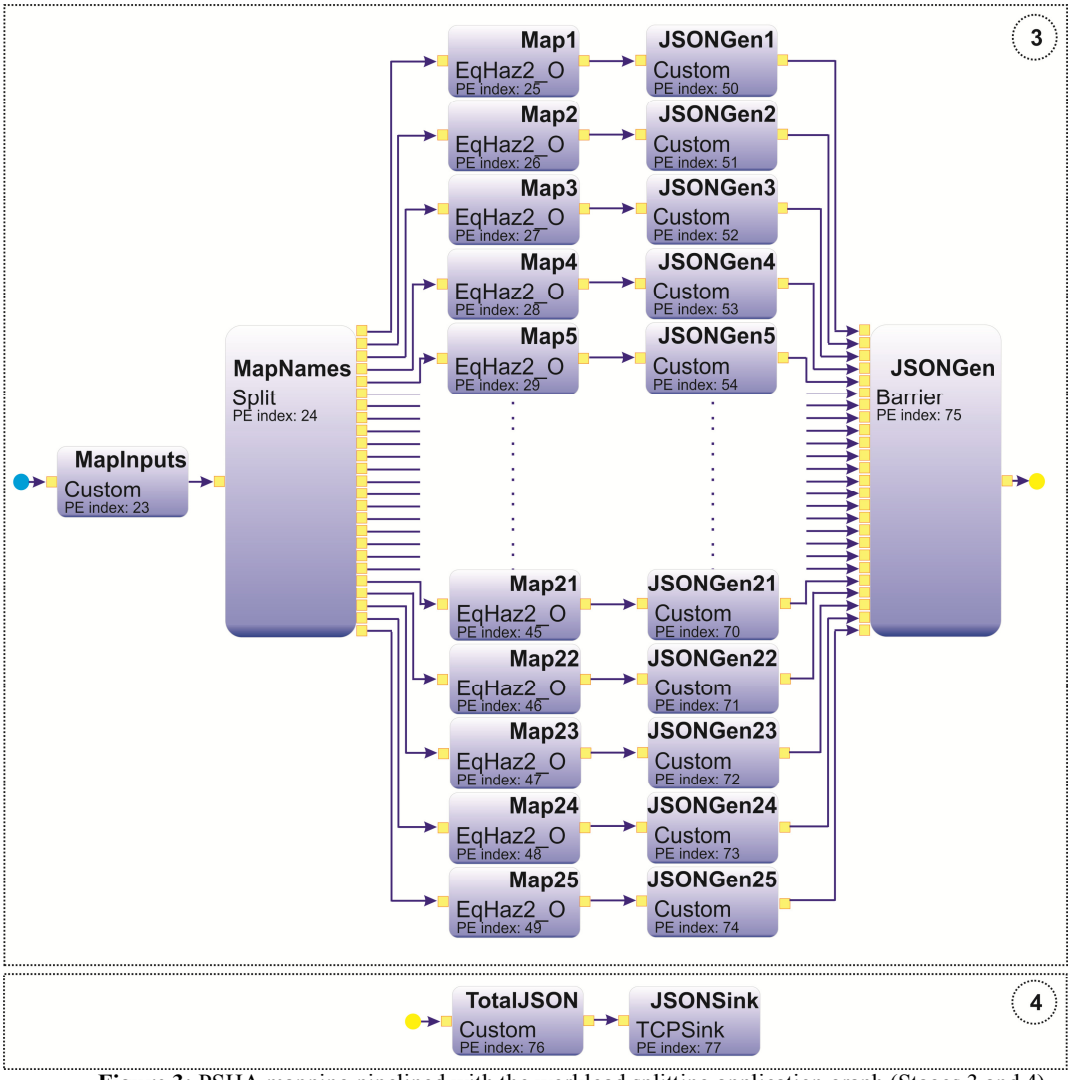


**Figure 3:** PSHA mapping pipelined with the workload splitting application graph (Stages 3 and 4)

The pipelined procedure is shown in Algorithm 2 in the Appendix. As a result, each single operator executes on a single core. The whole process is divided into four stages. The first stage is exactly the same as that shown in Figure 1, because the process of getting input parameters remains the same as before. The second stage (Figure 2) shows the decomposition and parallel implementation of Monte Carlo simulations for the synthetic catalogue. The third stage (Figure 3) presents the PEs performing spatial grid decomposition. Stage 4 of Figure 3 shows the final output. The connections between stages are shown with solid circles of the same color. That implementation provides an increase in the number of records in the synthetic catalog to as many as 10,000,000 records and the number of sites for each hazard map is stepped-up to 2,500,000 sites, allowing for the production of high resolution maps and significantly decreasing the execution time, as detailed below.

In Algorithm 2, the input stream for synthetic catalogue generation operator is divided into several streams in order to account for the EqHaz1 limit - the requested number of records in each catalogue has to be under 1,000,000 (Step 3). After the workload split is performed in (Step 3), each feeds into the catalogue generators to be processed in parallel (Steps 4-13) in order to optimize execution. To

satisfy the EqHaz1 limitations, the total number of records in the catalogue requested by the user should be less than the maximum number of records in each subcatalog multiplied by the number of parallel pipelines (10 in this example). When all subcatalogs are generated, the total synthetic catalog will be formed (Steps 14-15).

In order to overcome the EqHaz2 limit for the number of grid points, the input stream to the Map operator is split so that there are no more than 100,000 grid points for each Map operator (Step 18). In our case the entire grid is divided into 25 subgrids, where every subgrid contains less than 100,000 points. The EqHaz2 calculations for the ground motion acceleration parameters are performed on (Steps 19-43) and each output stream is converted to JSON format (Steps 44-69). Subsequently, results of all streams are merged together and sent to output port to be visualized.

# 4 Experimentation

The experimental environment in this work consists of a cluster of four machines, each with dual Xeon quad-core 2.4GHz CPU, 16GB RAM and running Linux. The computation power of the cluster is provided by the total of 32 cores. From the application point of view, at least 32 processes can be run in parallel on the cluster for handling heavy workload. InfoSphere Streams Version 3.2 has been configured and installed on the cluster. Files with input models (F1 and F2 contain GSC 2011 composite model of 39 Eastern Canada zones, F4 contain ground motion databases for Eastern Canada (Atkinson & Adams, 2013)) are available online from the official website of the engineering seismotoolbox (www.seismotoolbox.ca). To demonstrate the performance improvement by running multiple pipelines in parallel for PSHA mapping, the execution time has been measured for both implementations with the same processing workload. Figure 1 is for sequential processing (Seq.) without the workload splitting, while Figure 2 and Figure 3 are for running multiple pipelines in parallel (Paral.) on the cluster with the workload splitting. For the parallel experiment, 10 pipelines were employed at the catalogues generation stage and compiled into 10 PEs, and 25 pipelines were implemented at the map generation stage and compiled into 25 PEs. For the processing workload used in the experiment, the number of records in every synthetic catalog is 1,000,000 and the number of grid points is 100,000, because these are maximum limits for the sequential programs. To measure performance time, two additional operators were written to measure the execution time and output the results into a file: the first operator estimates the execution time of total synthetic catalog generation (Catalog); the second assesses the ground motion calculation performance time (Map). Timing results for Catalog and Map, in addition to the total execution time and the speedup (SF) factor, are summarized in the Table 2.

| Catalog (number of records = 1,000,000) | | | Map (number of grid points = 100,000) | | | Total | | |
|---|---|---|---|---|---|---|---|---|
| Seq. (s) | Para. (s) | SF (Seq./Para.) | Seq. (s) | Para. (s) | SF (Seq./Para.) | Seq. (s) | Para. (s) | SF (Seq./Para.) |
| 5.68 | 0.43 | 13.2 | 3780.3 | 138.2 | 27.4 | 3785.98 | 138.63 | 27.3 |

**Table 2:** Timing and speed results

The total speedup factor obtained is 27.3 and is the performance improvement with the pipelining implementation and paralleled executions. It should be noted that the performance increase is obtained not by the improvement of every primitive operator execution, but simply by subdivision of input streams and primitive operators pipelining.

An example of a final output map is shown in Figure 4. Red lines demonstrate a spatial decomposition of map grid. Each red rectangle contains 100,000 points and the total number of points is 2,500,000 for a map resolution equal to one value per kilometer.

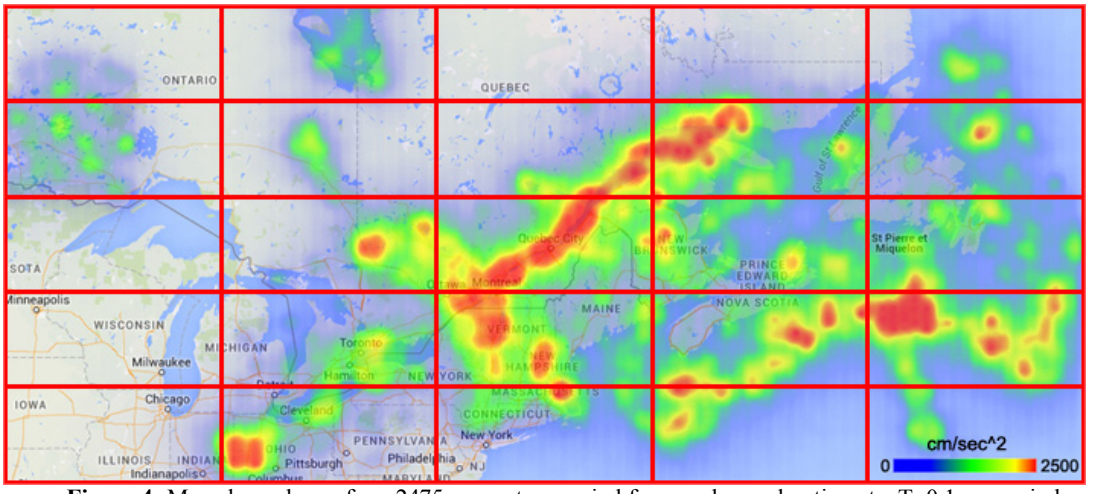


**Figure 4:** Mean hazard map for a 2475 year return period for pseudo acceleration at a T=0.1 sec period

The total procedure execution time is 5.6 minutes. This example demonstrates a significant improvement in PSHA mapping procedure. For comparison, Assatourians and Atkinson (2013) produced hazard maps with a resolution of 3326 points created by the EqHaz suite. The PSHA procedure execution time for that case was approximately 90 minutes (Assatourians & Atkinson, 2013) with the experimental environment consists of one machine with dual quad-core 2.4 GHz CPU, 8 Gb RAM and running Windows 7 Professional 64-bit (personal communications, K. Assatourians).

# 5 Conclusions

Seismic hazard maps have many practical and important applications and are widely used by policymakers to assign earthquake-resistant construction standards, by insurance companies to set insurance rates, by civil engineers to estimate structural stability and by emergency hazard agencies to estimate hazard potential and the associated response. Therefore, the results of this work are important for emergency hazard providers, demonstrating that the use of the Streams platform make it possible to produce high resolution hazard maps in near-real time. Here we have provided an example for Eastern Canada, but the method can be applied to any region where PSHA input parameters are available. The pipelining implementation is flexible and scalable for extension and deployment onto a larger cluster. As a result, it is possible to obtain even higher resolution with better performance time.

In the future, this work could be extended to achieve increased performance from additional decomposition on two levels: synthetic catalogs and map generation stages, with even greater cluster support. Additional performance tests should be done to determine the optimum number of parallel pipelines. In the long term, this work could be used as a basis for a universal PSHA streams toolkit development.

# Acknowledgements

Development of this tool has been made possible due to the joint efforts of the Western University Computational Laboratory for Fault System Modeling, Analysis, and Data Assimilation and the consortium of Canadian academic institutions, a high performance computing network SHARCNET. We are grateful to Dr. K. Assatourians for his assistance with EqHaz. The system object libraries are compiled using the Intel® Fortran Compiler, providing an opportunity to adapt the EqHaz FORTRAN code with no major modifications.

# References

Assatourians, K. & Atkinson, G. M., 2013. EqHaz: An open-source probabilistic seismic-hazard code based on the Monte Carlo simulation approach.. *Seismol. Res. Lett.,* 84(3), pp. 516-524.

Atkinson, G. M. & Adams, J., 2013. Ground motion prediction equations for application to the 2015 national seismic hazard maps of Canada. *Can. J. Civil Eng.,* Volume 40, pp. 988-998.

Bauer, M. A., Biem, A., McIntyre, S. & Xie, Y., 2010. *A Pipelining Implementation for Parsing X-ray Diffraction Source Data and Removing the Background Noise.* s.l., J. Phys.: Conf. Ser..

Bender, B. & Perkins, D. M., 1982. SEISRISK II: A computer program for seismic hazard estimation. *Open-File Report,* pp. 82-293.

Bender, B. & Perkins, D. M., 1987. SEISRISK III: A computer program for seismic hazard estimation. *U.S. Geol. Surv. Bull. ,* Volume 1772.

Cornell, C. A., 1968. Engineering seismic risk analysis. *Bull. Seismol. Soc. Am.,* 58(5), pp. 1583-1606.

Field, N., Jordan, T. A. & Cornell, C. A., 2003. OpenSHA: A developing community-modeling environment for seismic hazard analysis. *Seismol. Res. Lett.,* 74(4), pp. 406-419.

IBM, 2014. *IBM Knowledge Center.* [Online] Available at: http://www-01.ibm.com/support/knowledgecenter/

McGuire, R., 1976. Fortran program for seismic risk analysis. *U.S. Geol. Surv. Open-File Rept.,* pp. 76-67.

McGuire, R., 1978. FRISK: Computer program for seismic risk analysis using faults as earthquake sources. *U.S. Geol. Surv. Open-File Rept.,* pp. 78-1007.

McGuire, R., 2004. *Seismic Hazard and Risk Analysis.*. Oakland, California.: Earthquake Engineering Research Institute.

Musson, R., 1999. Determination of design earthquakes in seismic hazard analysis through Monte Carlo simulation. *J.Earthquake Eng.,* Issue 3, pp. 463-474.

Musson, R., 2000. The use of Monte Carlo simulations for seismic hazard assessment in the U.K.. *Ann.Geofis.,* Issue 43, pp. 1-9.

Musson, R. & Henni, P., 2001. Methodological considerations of probabilistic seismic hazard mapping. *Soil Dynamics and Earthquake Engineering,* Issue 21, pp. 385-403.

Ordaz, M., Martinelli, F., D'Amico, V. & Meletti, C., 2013. CRISIS2008: A flexible tool to perform probabilistic seismic hazard assessment. *Seismol. Res. Lett.,* 84(3), pp. 495-504.

Robinson, D., Dhu, T. & Schneider, J., 2006. Practical probabilistic seismic risk analysis: A demonstration of capability. *Seismol. Res. Lett.,* 77(4), pp. 453-459.

Robinson, D., Fulford, G. & Dhu, T., 2005. EQRM: Geoscience Australia's Earthquake Risk Model: Technical Manual: Version 3.0. *GA Record 2005/01, Geoscience Australia, Canberra,* p. 148.

Wied, P., 2014. *Dynamic Heatmaps for the Web.* [Online] Available at: http://www.patrick-wied.at/static/heatmapjs/

# 6 Appendix

**Input** : File F1 with the description of seismic source zone model from www.seismotoolbox.ca, file F2 with a list of zones (polygon coordinates and seismicity re-occurrence parameters from the Geological Survey of Canada (GSC)), file F3 with other PSHA parameters specified by user, file F4 contains GMPE model
**Output**: File J encoding map in JSON format, containing ground motion estimation for a specified number of sites

```
1   N1 ← Barrier(F1,F2,F3);
2   I1 ← CatalogInputs(N1);
3   G ← Catalog(I1);
4   N2 ← Barrier(F3, G, F4);
5   I2 ← MapInputs(N2);
6   O ← Map(I2);
7   J ← JSONGen(O);
8   TCPSink(J);
```

**Algorithm 1:** Production of hazard map

**Input** : File F1 with the description of seismic source zone model from www.seismotoolbox.ca, file F2 with a list of zones (polygon coordinates and seismicity re-occurrence parameters from the GSC), file F3 with other PSHA parameters specified by user, file F4 contains the GMPE model
**Output**: File JS encoding map in JSON format containing a ground motion estimation for a specified number of sites

```
1   N1 ← Barrier(F1,F2,F3);
2   I1 ← CatalogInputs(N1);
3   C1, C2, ..., C10 ← Split(I1);
4   G1 ← Catalog(C1);
5   G2 ← Catalog(C2);
    ...
13  G10 ← Catalog(C10);
14  T ← Barrier(G1, G2, ..., G10);
15  U ← TotalCatalog(T);
16  N2 ← Barrier(F3, U, F4);
17  I2 ← MapInputs(N2);
18  M1, M2, ..., M25 ← Split(I2);
19  O1 ← Map(M1);
20  O2 ← Map(M2);
    ...
43  O25 ← Map(M25);
44  J1 ← JSONGen(O1);
45  J2 ← JSONGen(O2);
    ...
68  J25 ← JSONGen(O25);
69  TJ ← Barrier(J1, J2, ..., J25);
70  JS ← TotalJSON(TJ);
71  TCPSink(JS);
```

**Algorithm 2:** Production of hazard map with parallel execution